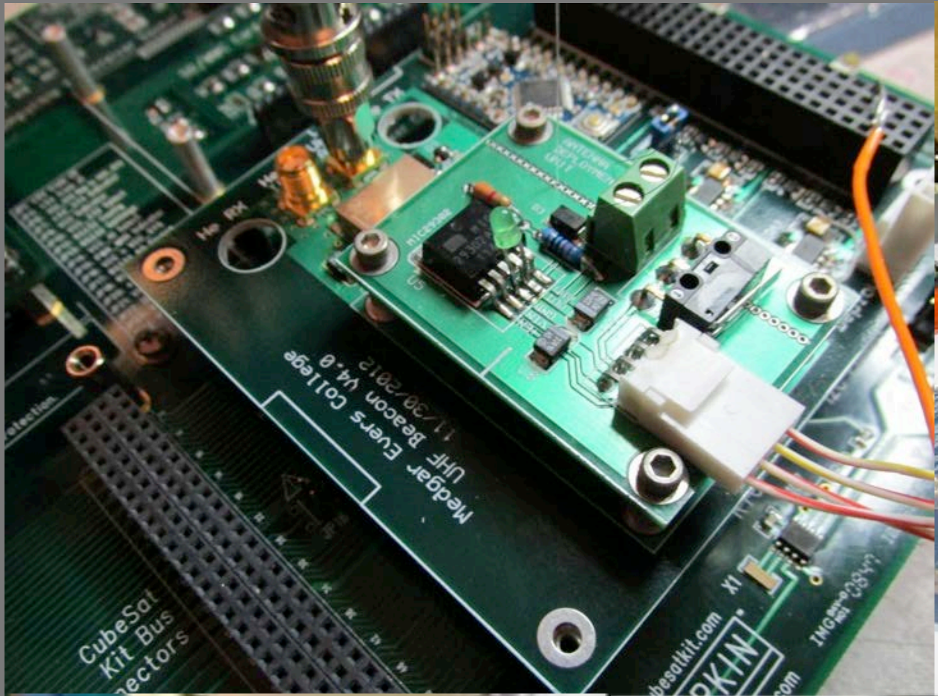
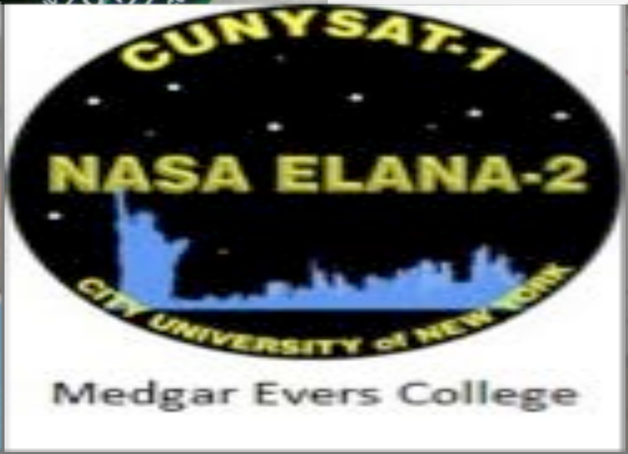
Intelligent Software for Nanosatellites

Interns: Jarmal Johnny, Edouard Michel, Valina Maitland, Tania Nelzy
Faculty Mentors: Dr. Shermane Austin, Prof. Michael Jiang





Abstract

NASA's CubeSat launch initiative provides opportunities for students to get exposure to NASA activities and gain interest in STEM careers. Cubesats are nanosatellites and software development plays an important role in control. The software for controlling, tracking, and communicating with the satellite has to be developed following best practices and ensuring that it is robust and able to tolerate and recover from the unexpected.

CubeSat

- Developed in 1999 by Prof. Jordi PuigSuari at California Polytechnic State University, Sans Luis Obispo, and Prof. Bob Twiggs at Stanford University’s Space Systems Development Library with the intention to provide a standard of micro and picosatellites.
- Sizes range from 1U, 10cm cubes with scale factors for 2U to 6U (presently).
- Low-cost, low-risk missions for education training, technology testing and scientific investigation – colleges and universities, federal agencies, e.g. NASA and DOD, and aerospace industry.

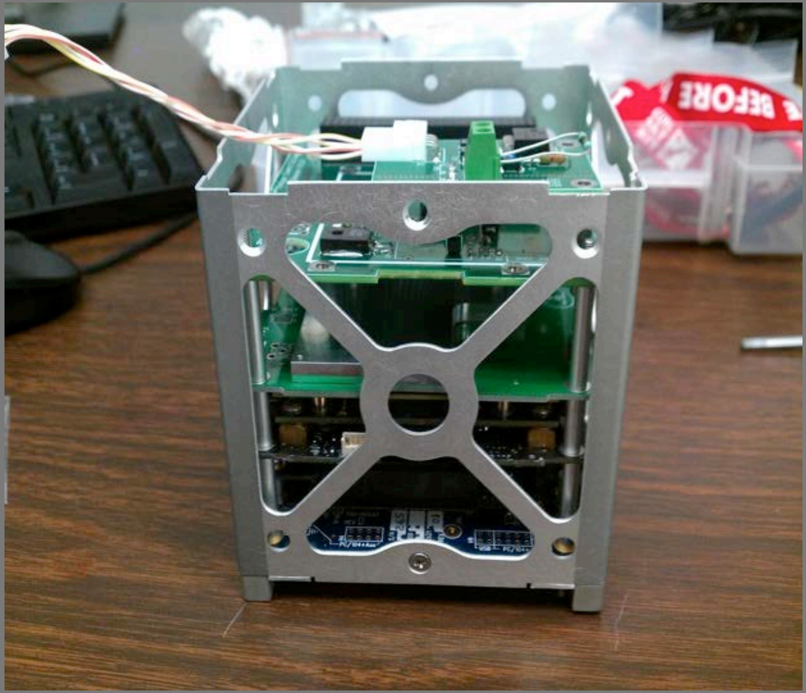
CUNYSAT-1

- Launched on December 5, 2013 on an Atlas V rocket
- Objectives were:

  -To design, assemble and test a basic pathfinder with mostly commercial-off-the-shelf components.

  - Provide workforce experiences for undergraduate students.

  - To build capacity for faculty research.



Autonomic Computing

Autonomic computing is a computer model that automatically controls the functions of computer applications and systems without the need of outside input from the user.

It is inspired by the human body’s nervous system which responds automatically to stimuli without conscious effort.

Computer systems are getting more and more complex so the goal is to have computers run themselves.

Elements of Autonomic Computing

The system must know its capabilities and limitations.

The system must be able to automatically configure and reconfigure itself.

The system must be self-optimizing.

The system must be able to work around problems.

Elements of Autonomic Computing

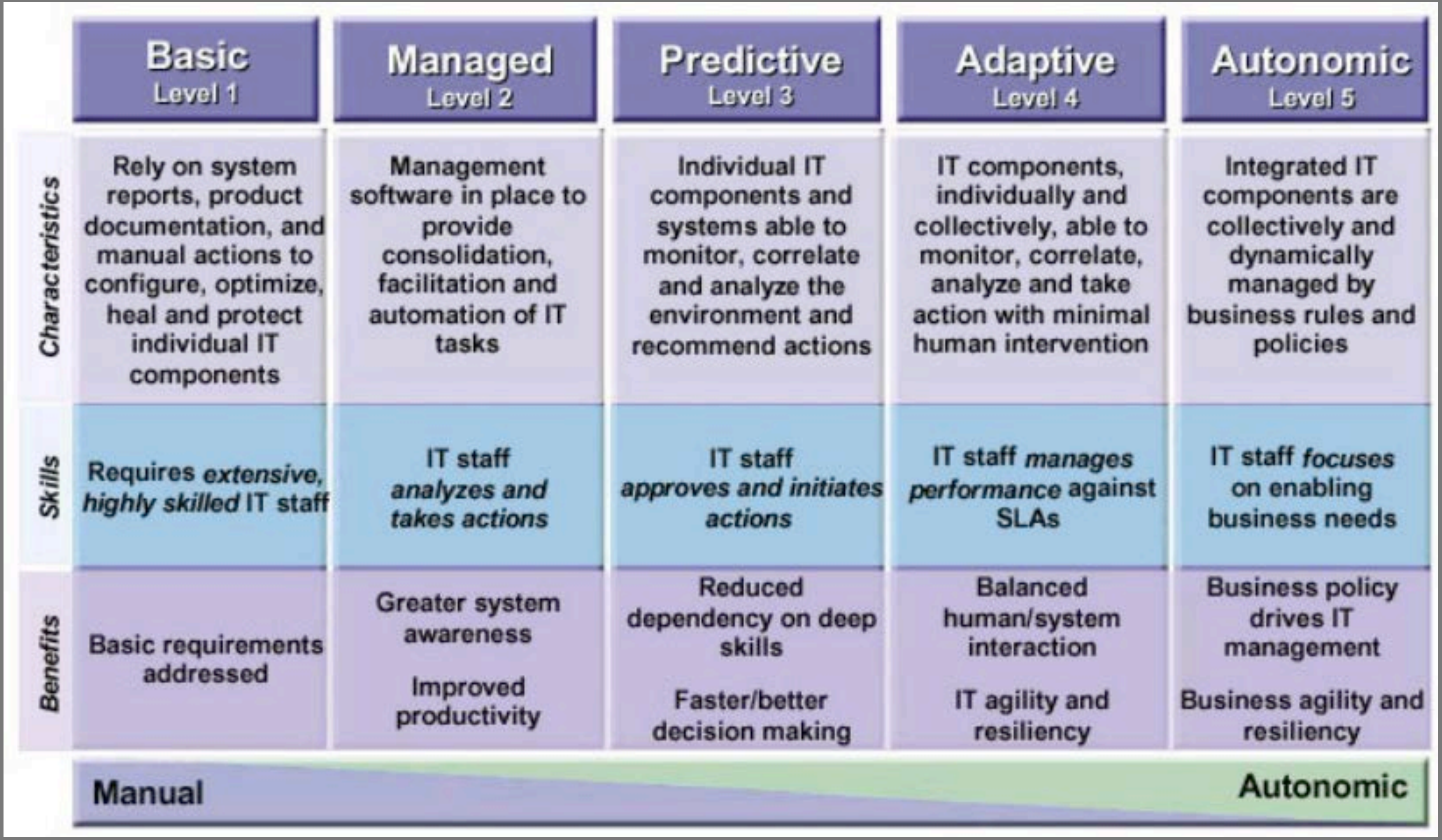
The system must be able to detect and protect itself from threats.

The system must be able to adapt to its environment.

The system must rely on open standards.

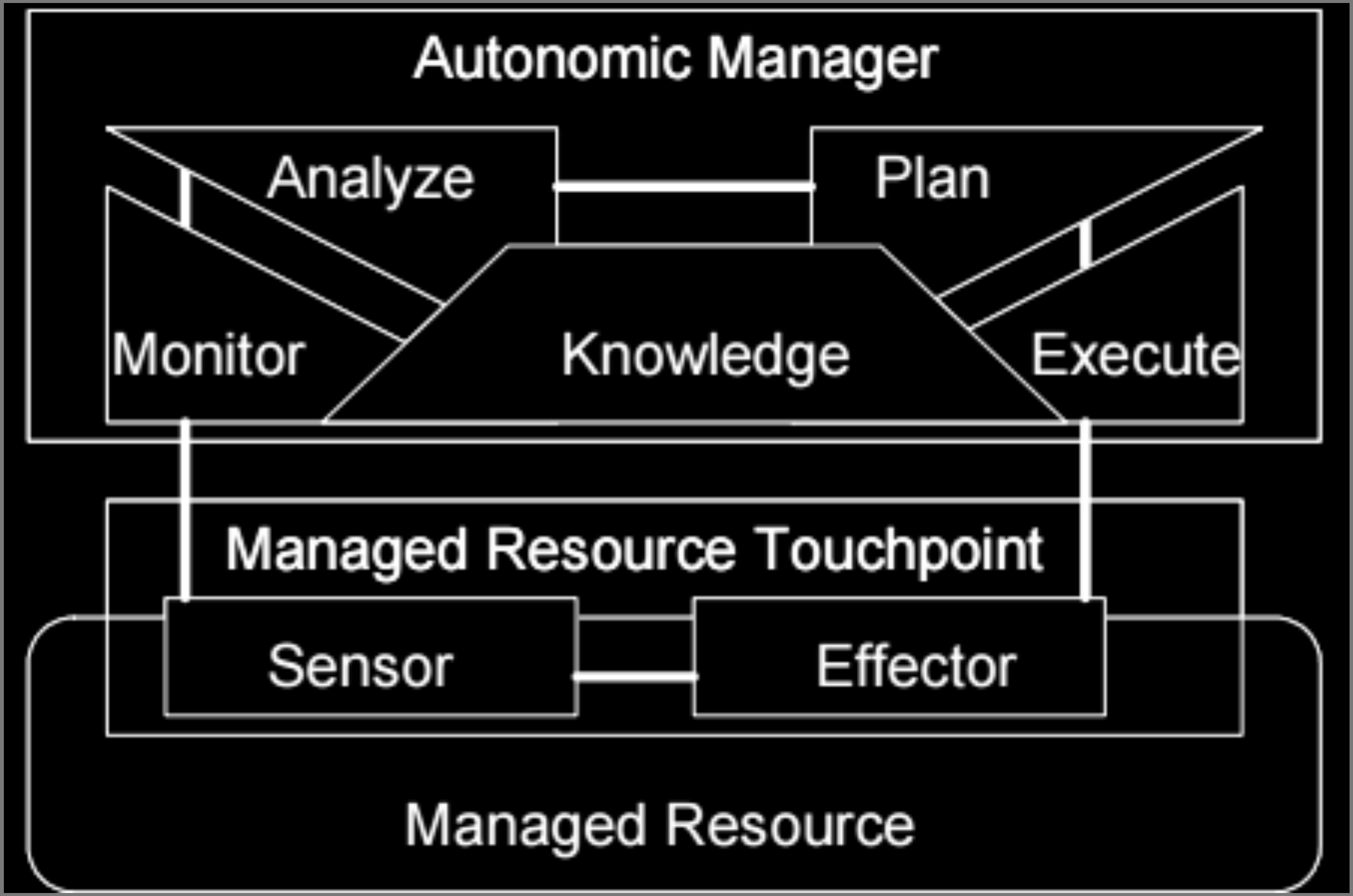
The system must anticipate demand on resources.

Levels of Autonomic Computing

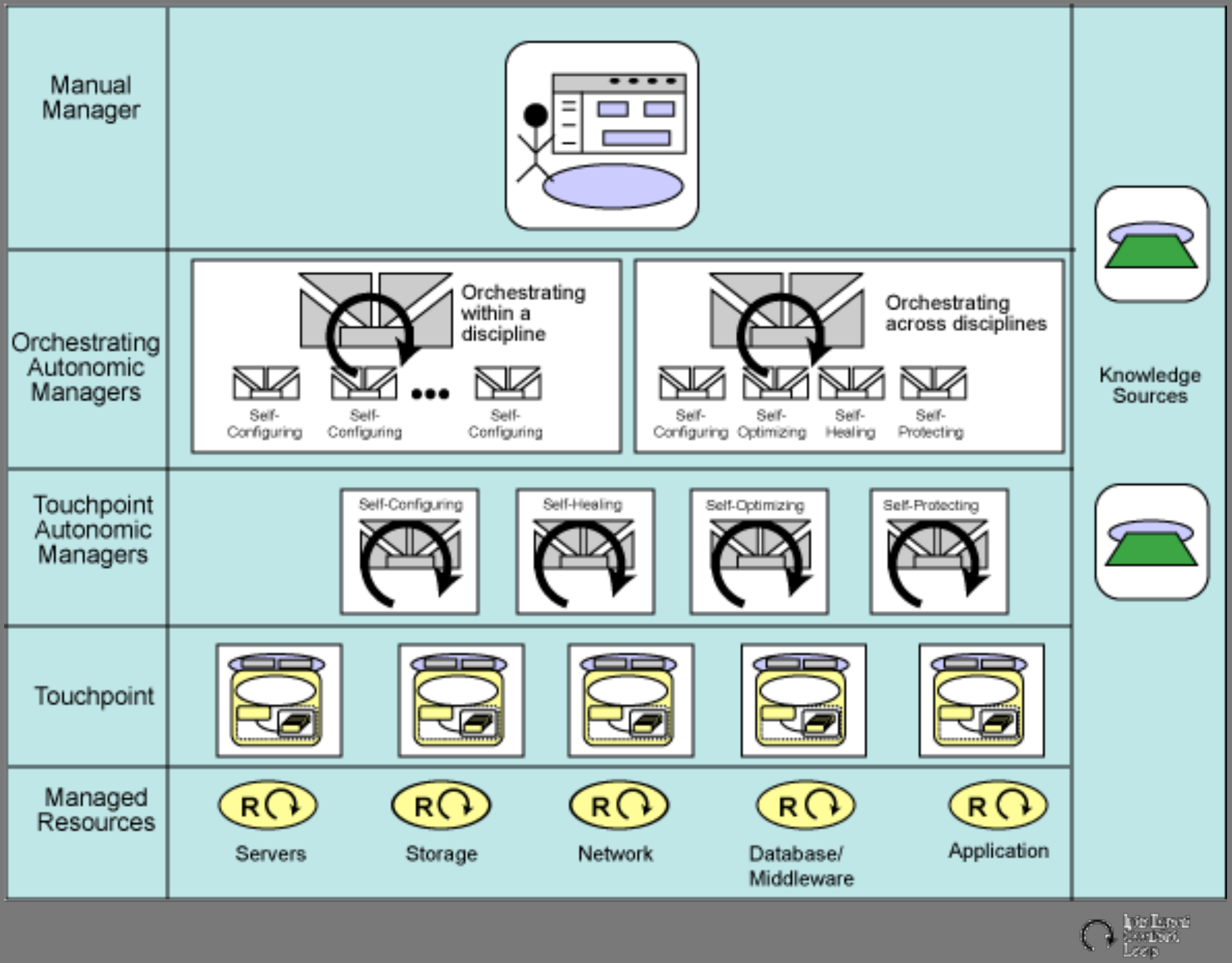


	Basic Level 1	Managed Level 2	Predictive Level 3	Adaptive Level 4	Autonomic Level 5
Characteristics	Rely on system reports, product documentation, and manual actions to configure, optimize, heal and protect individual IT components	Management software in place to provide consolidation, facilitation and automation of IT tasks	Individual IT components and systems able to monitor, correlate and analyze the environment and recommend actions	IT components, individually and collectively, able to monitor, correlate, analyze and take action with minimal human intervention	Integrated IT components are collectively and dynamically managed by business rules and policies
Skills	Requires *extensive, highly skilled* IT staff	IT staff *analyzes and takes actions*	IT staff *approves and initiates actions*	IT staff *manages performance* against SLAs	IT staff *focuses* on enabling business needs
Benefits	Basic requirements addressed	Greater system awareness Improved productivity	Reduced dependency on deep skills Faster/better decision making	Balanced human/system interaction IT agility and resiliency	Business policy drives IT management Business agility and resiliency
	Manual				Autonomic

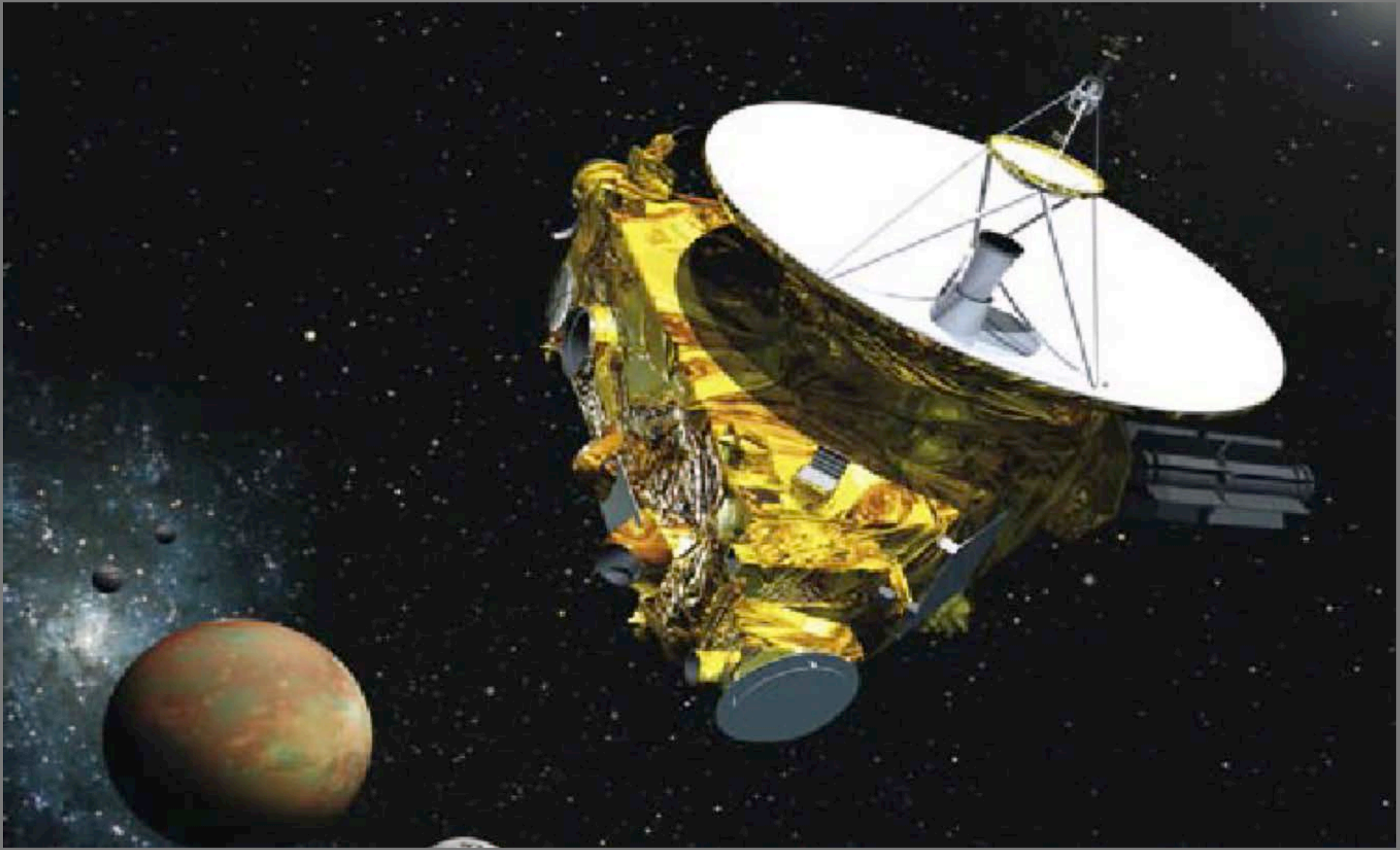
Autonomic Computing - Control Loop



Autonomic Computing - Overall System



New Horizons



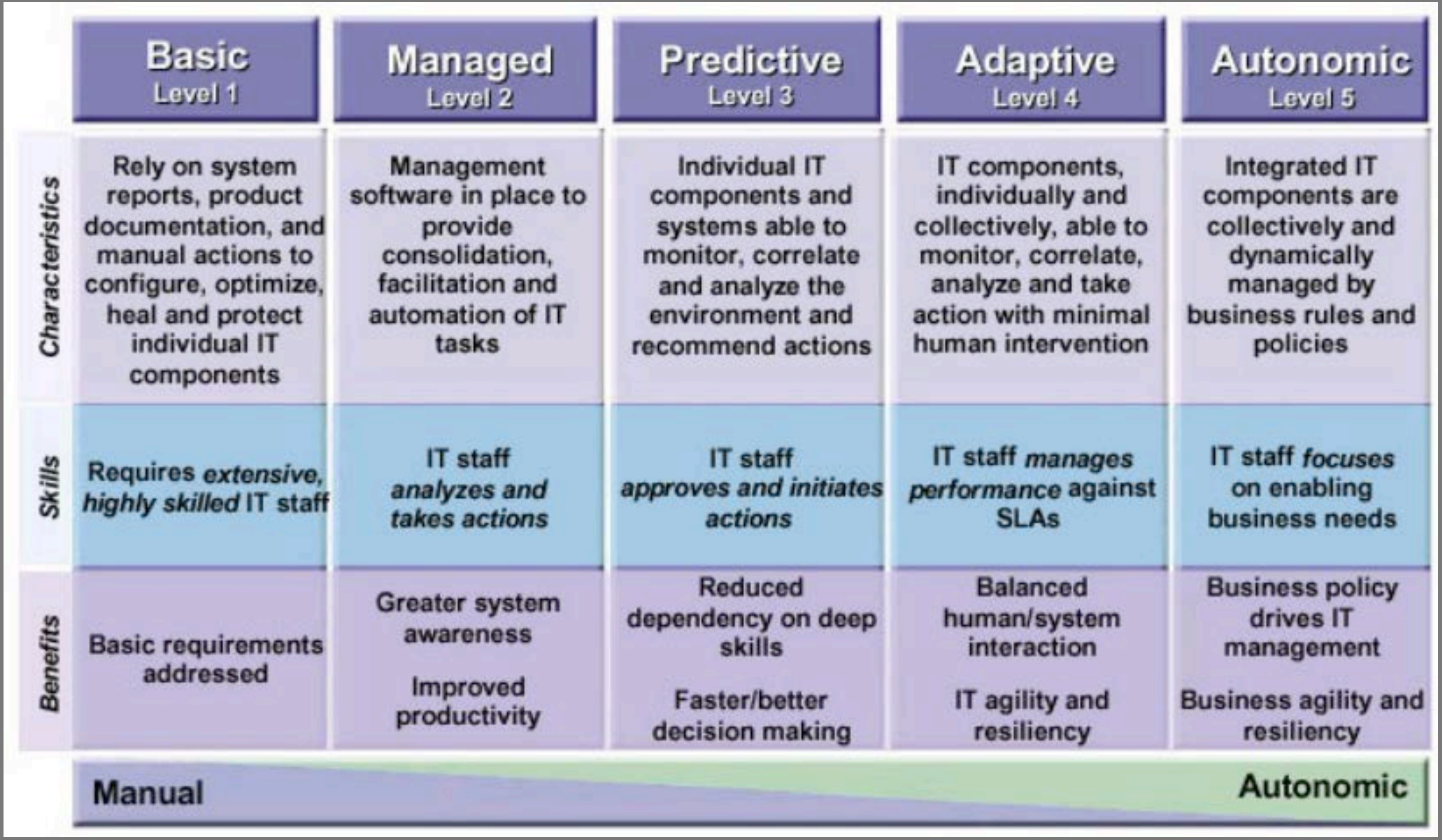
New Horizons

On July 4th New Horizon temporarily lost contact with Earth for a few minutes

The computer tried two intensive calculations at once which was more than it could handle.

The computer autonomously put itself in safe-mode to protect itself and the mission, and commanded itself to reestablish communication with Earth.

Levels of Autonomic Computing



	Basic Level 1	Managed Level 2	Predictive Level 3	Adaptive Level 4	Autonomic Level 5
Characteristics	Rely on system reports, product documentation, and manual actions to configure, optimize, heal and protect individual IT components	Management software in place to provide consolidation, facilitation and automation of IT tasks	Individual IT components and systems able to monitor, correlate and analyze the environment and recommend actions	IT components, individually and collectively, able to monitor, correlate, analyze and take action with minimal human intervention	Integrated IT components are collectively and dynamically managed by business rules and policies
Skills	Requires *extensive, highly skilled* IT staff	IT staff *analyzes and takes actions*	IT staff *approves and initiates actions*	IT staff *manages performance* against SLAs	IT staff *focuses* on enabling business needs
Benefits	Basic requirements addressed	Greater system awareness Improved productivity	Reduced dependency on deep skills Faster/better decision making	Balanced human/system interaction IT agility and resiliency	Business policy drives IT management Business agility and resiliency

Manual → Autonomic

Why We Study Electronics

In order to successfully and efficiently develop appropriate software, we need to have a basic understanding of electronics.

Terminology

VIN: Voltage in

GND: Ground (a common return path for the electrical current)

INT: Only needed when you have a sensor configured to signal a change

ADC: Analog to Digital converter. Takes analog information as an input and converts it to a digital output.

TX: Transmit Line is used to send data.

RX: Receive Line is used to receive data.

SDA: Serial Data Line is used to communicate. Hardware can receive and transmit on this line.

SCL: Serial Clock Line is used to synchronize communication between the devices.

Terminology (Continued)

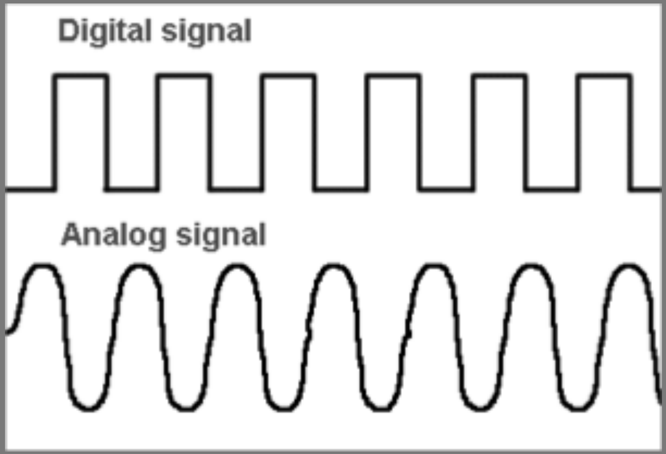
In **asynchronous** communication the devices must know in advance the rate of communication. It is the simplest form of communication between devices.

I2C is a communication protocol that is capable of communicating with multiple devices simultaneously. It is more complicated but more reliable than serial.

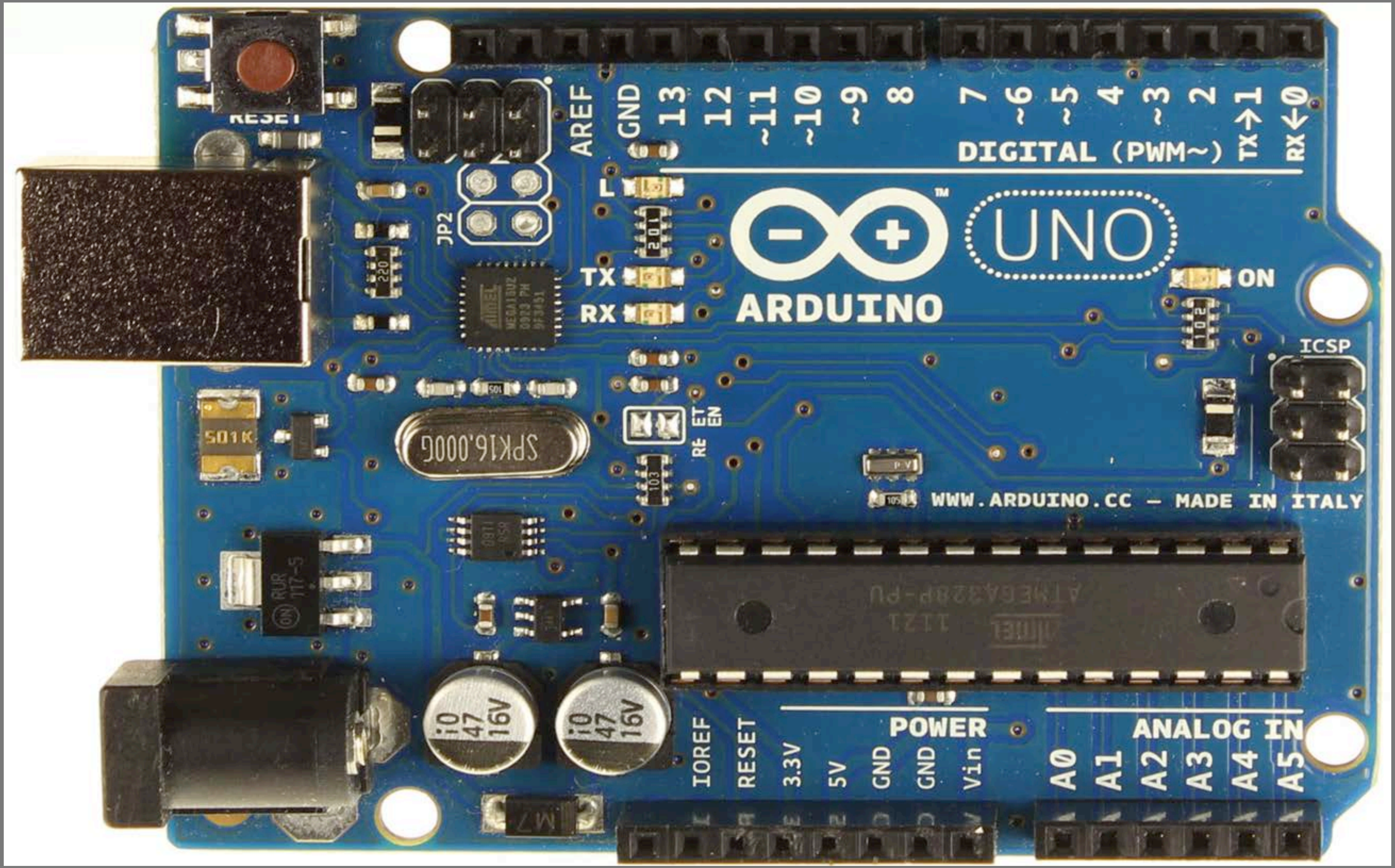
Digital vs. Analog

Digital information is discrete and discontinuous. It holds a finite range of values. For our digital pins these are on and off which is represented by high/low or voltage/ground.

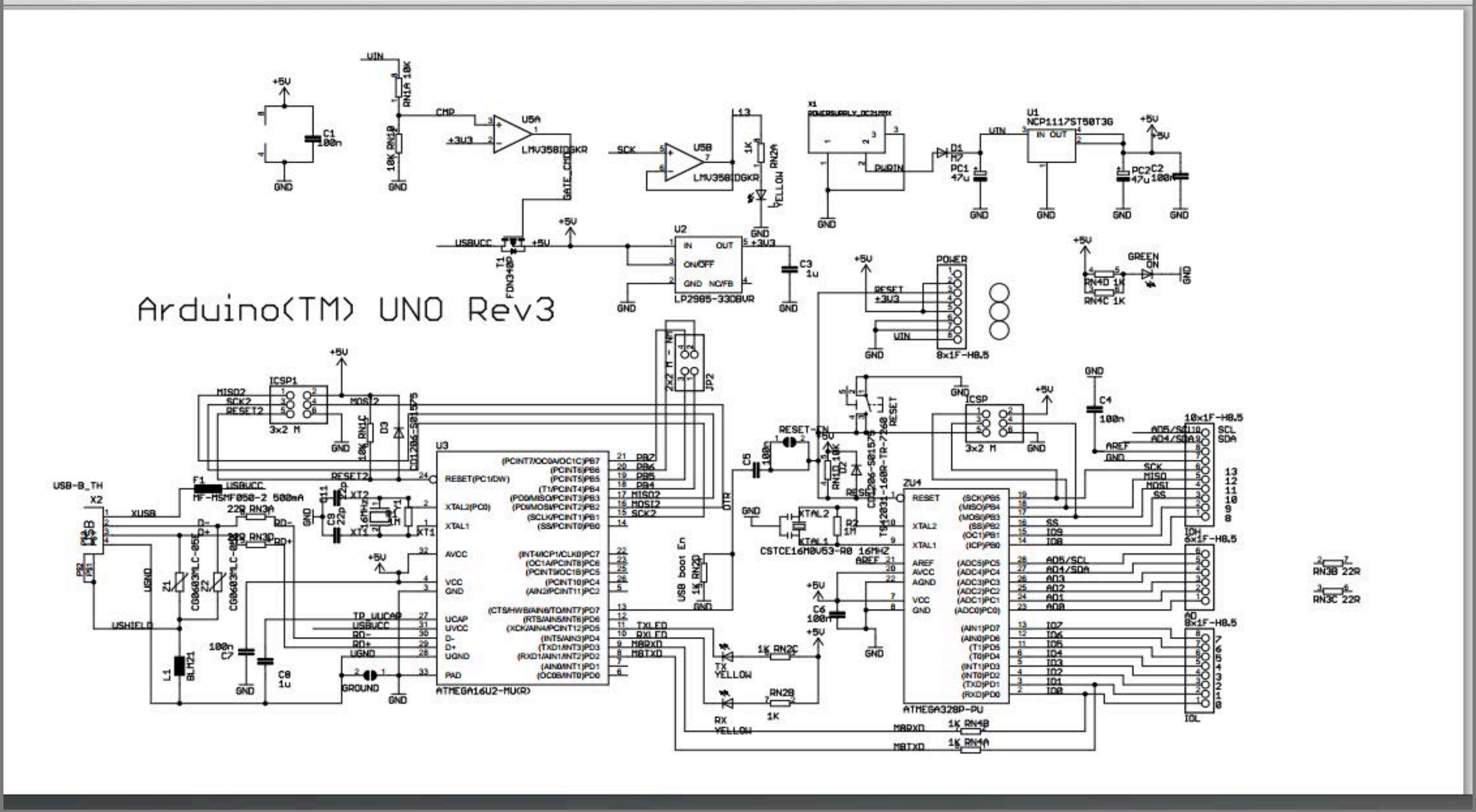
Analog is a way of representing information that uses a continuous range of values.



Microcontroller: Arduino



Arduino



Arduino(TM) UNO Rev3
ATMEGA16U2-MU(R)
ATMEGA328P-PU
USB-B_TH
ICSP1
ICSP
POWER
LP2985-33DBVR
NCP1117ST50T3G
LMV358IDGKR
RESET

Arduino UNO R3

Microcontroller: ATmega328P

Clock Speed: 16 Mhz
Flash memory: 32 kB
SRAM: 2 kB
EEPROM: 1 kB
Operating Voltage: 5V

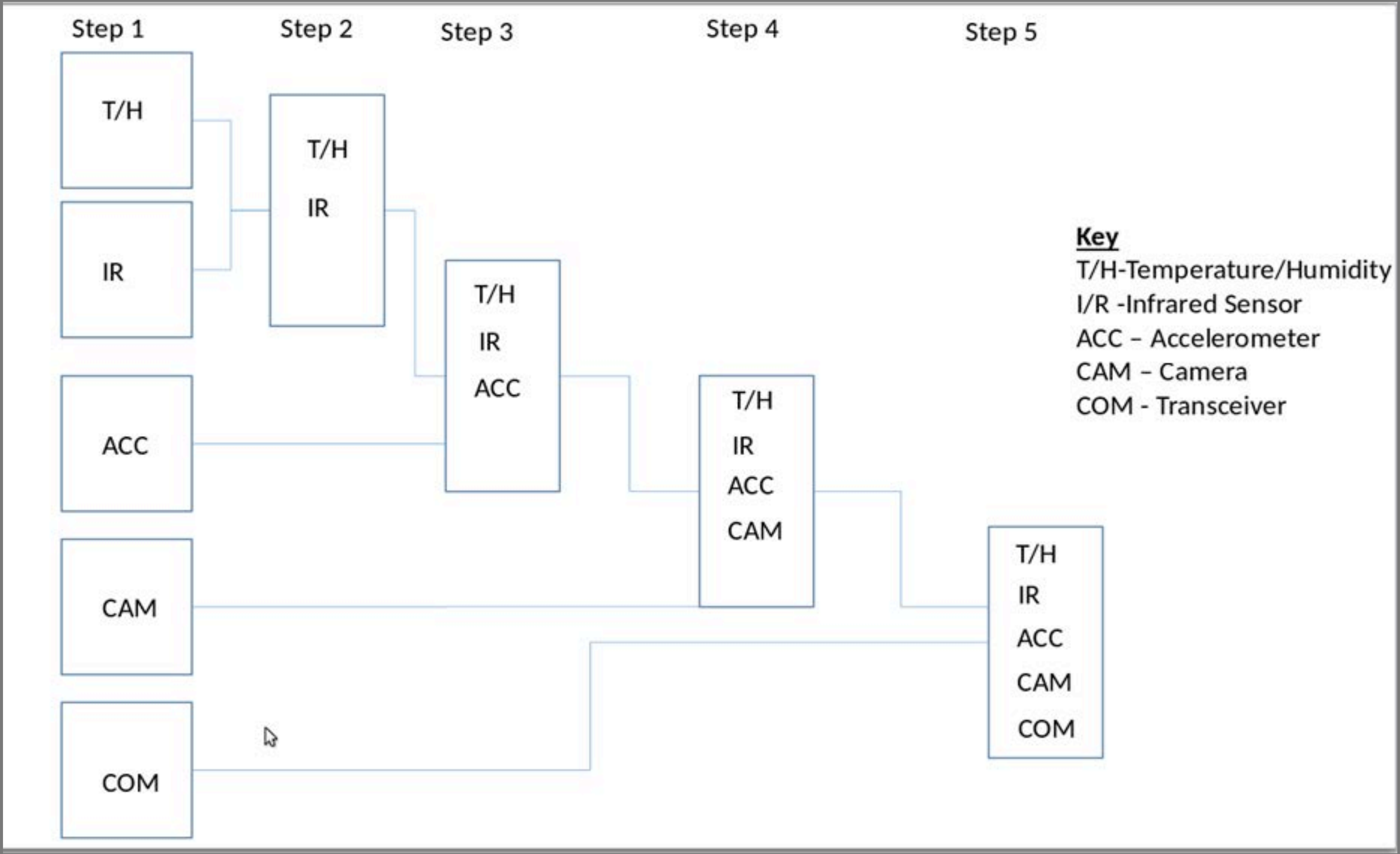
Digital I/O Pins: 14 (6 are PWM)
Analog Input Pins: 6

Arduino

Arduino is an open source tool utilized for developing electronic devices. It includes a microcontroller board with writing software to create and test codes.

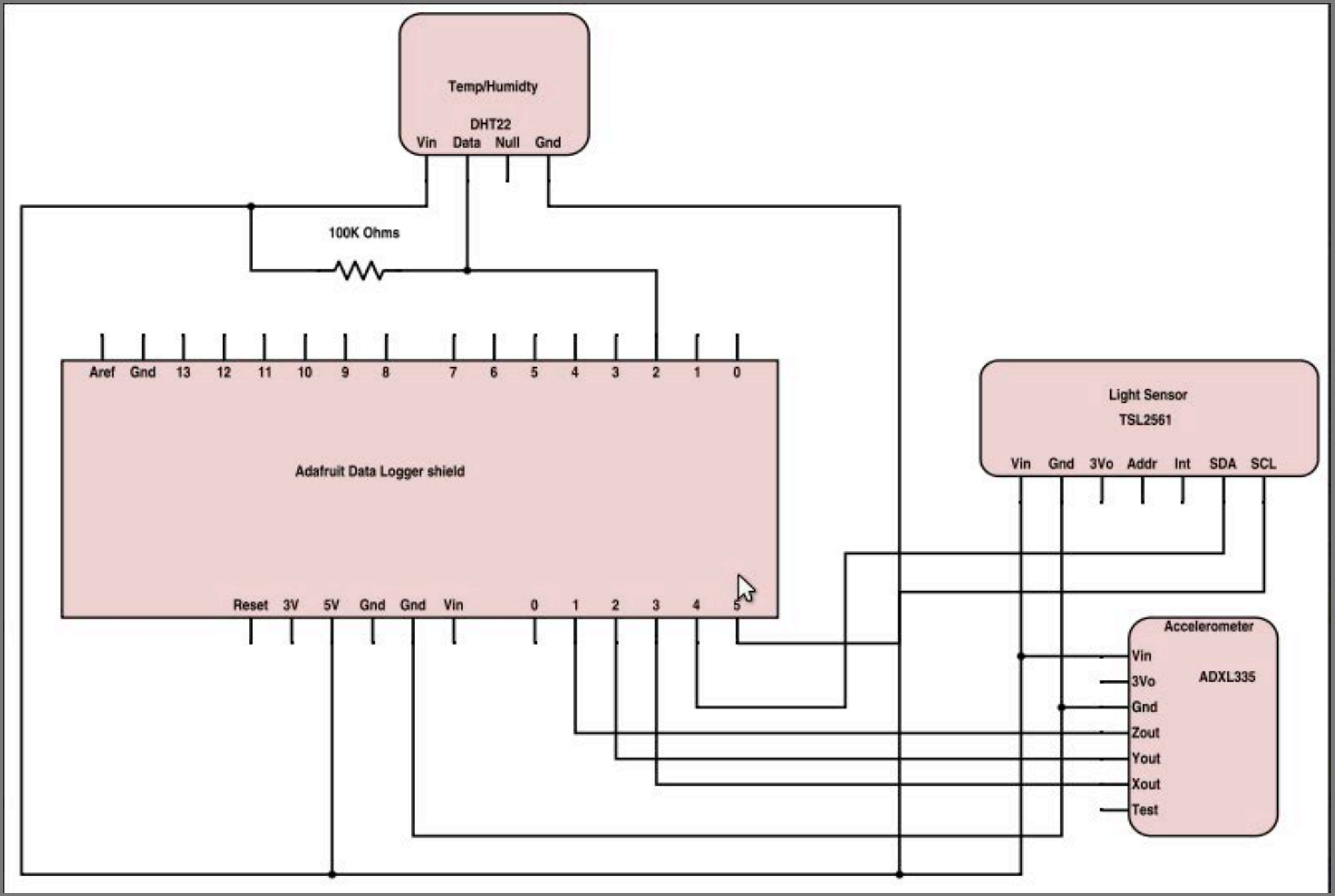
Arduino can be used to input data from sensors, control physical outputs, and communicate with computer software.

Integration Process

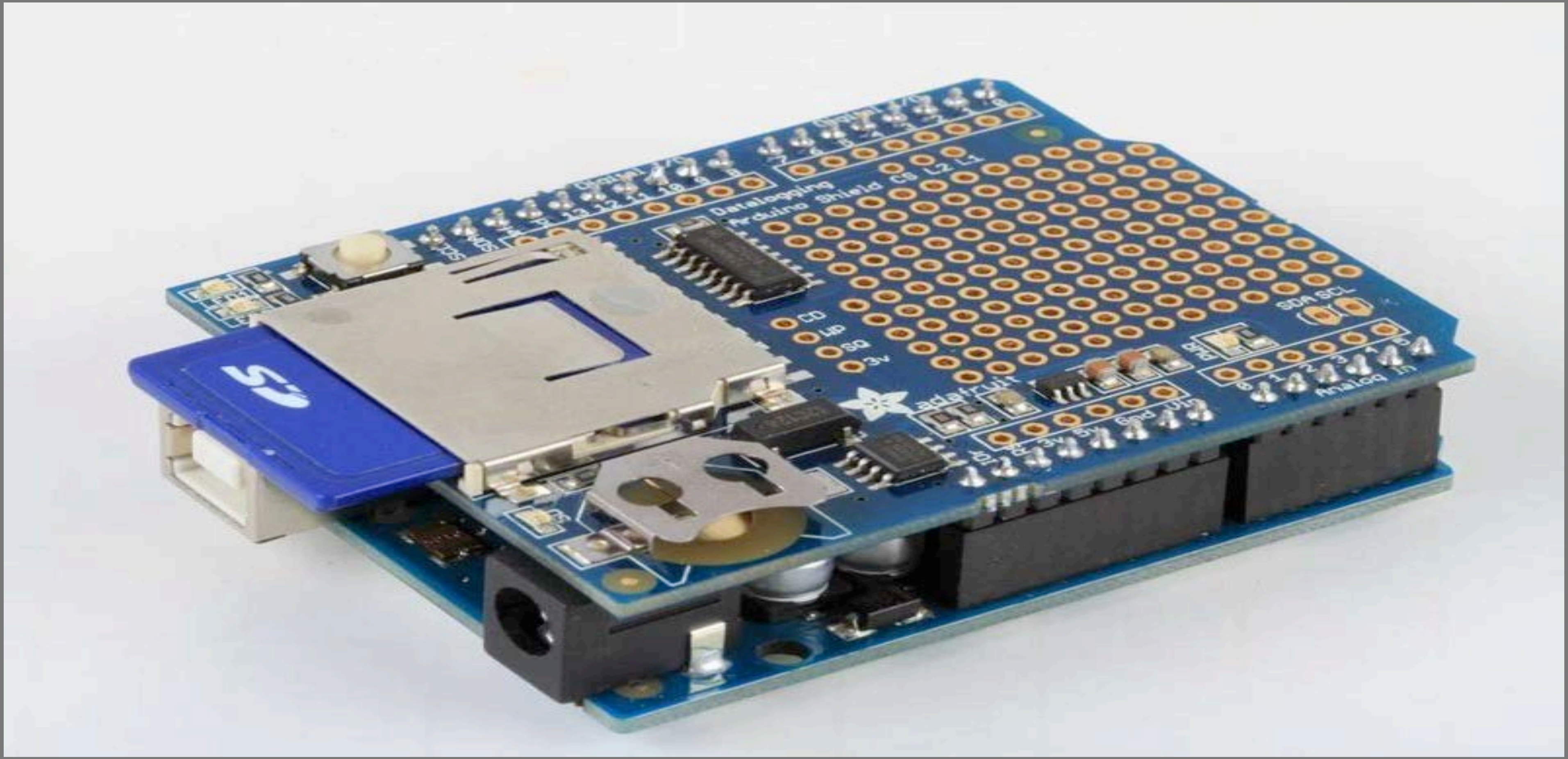


Key
T/H-Temperature/Humidity
I/R -Infrared Sensor
ACC – Accelerometer
CAM – Camera
COM - Transceiver

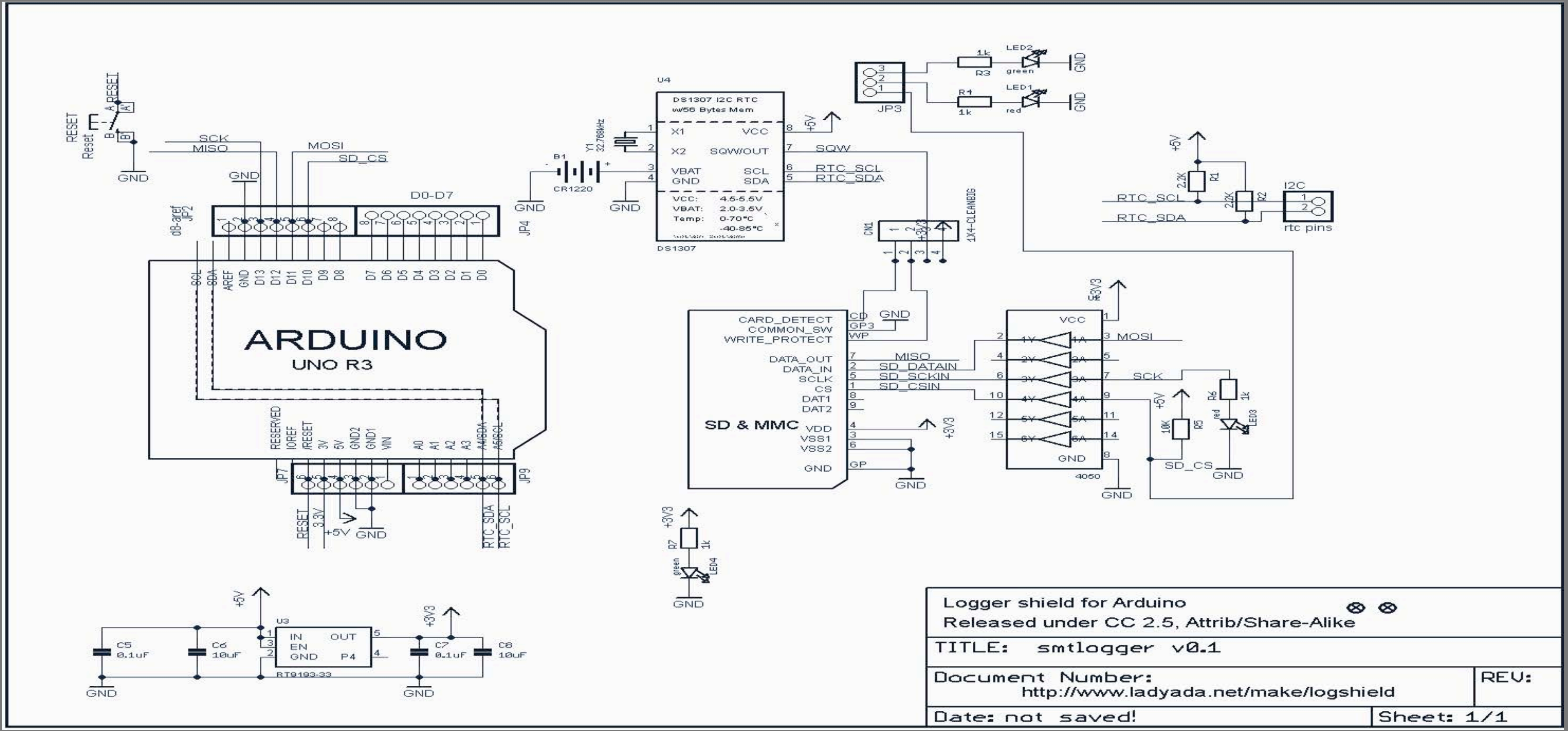
Schematic


Temp/Humidty
DHT22
Vin
Data
Null
Gnd
100K Ohms
Aref
Gnd
13
12
11
10
9
8
7
6
5
4
3
2
1
0
Adafruit Data Logger shield
Reset
3V
5V
Gnd
Gnd
Vin
0
1
2
3
4
5
Light Sensor
TSL2561
Vin
Gnd
3Vo
Addr
Int
SDA
SCL
Accelerometer
ADXL335
Vin
3Vo
Gnd
Zout
Yout
Xout
Test

Data Shield


Datalogging
Arduino Shield
Digital I/O
Analog In
SDA SCL
adafruit

Data Shield


ARDUINO
UNO R3
DS1307 I2C RTC
SD & MMC
Logger shield for Arduino
Released under CC 2.5, Attrib/Share-Alike
TITLE: smtlogger v0.1
Document Number: http://www.ladyada.net/make/logshield
REV:
Date: not saved!
Sheet: 1/1

It is an attachment for the Arduino that allows us to interface with an SD card and a real time clock without using up any pins.

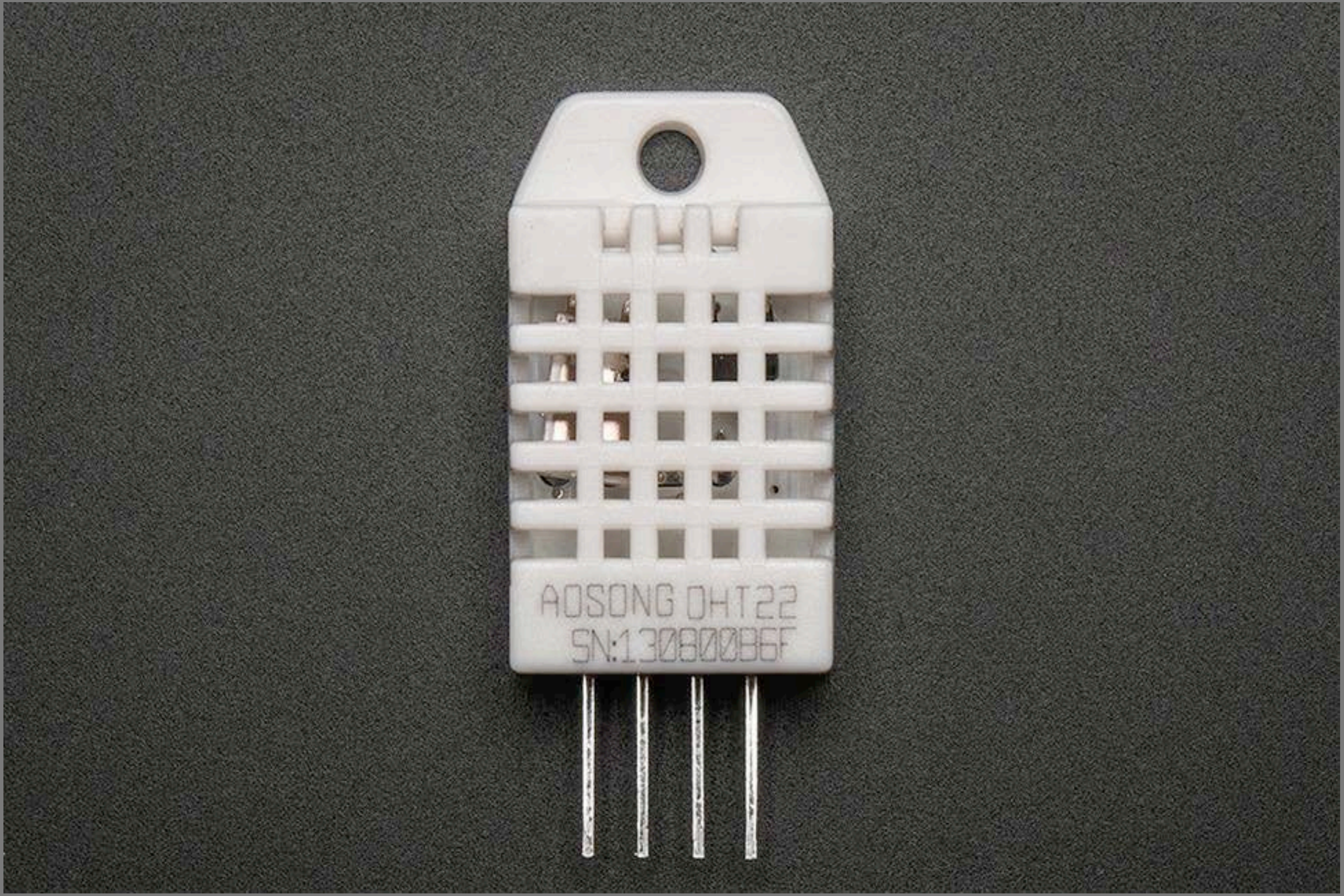
SD Card

SanDisk
microSD
microSDHC
microSDXC
ADAPTER
SanDisk
8GB
micro SD HC

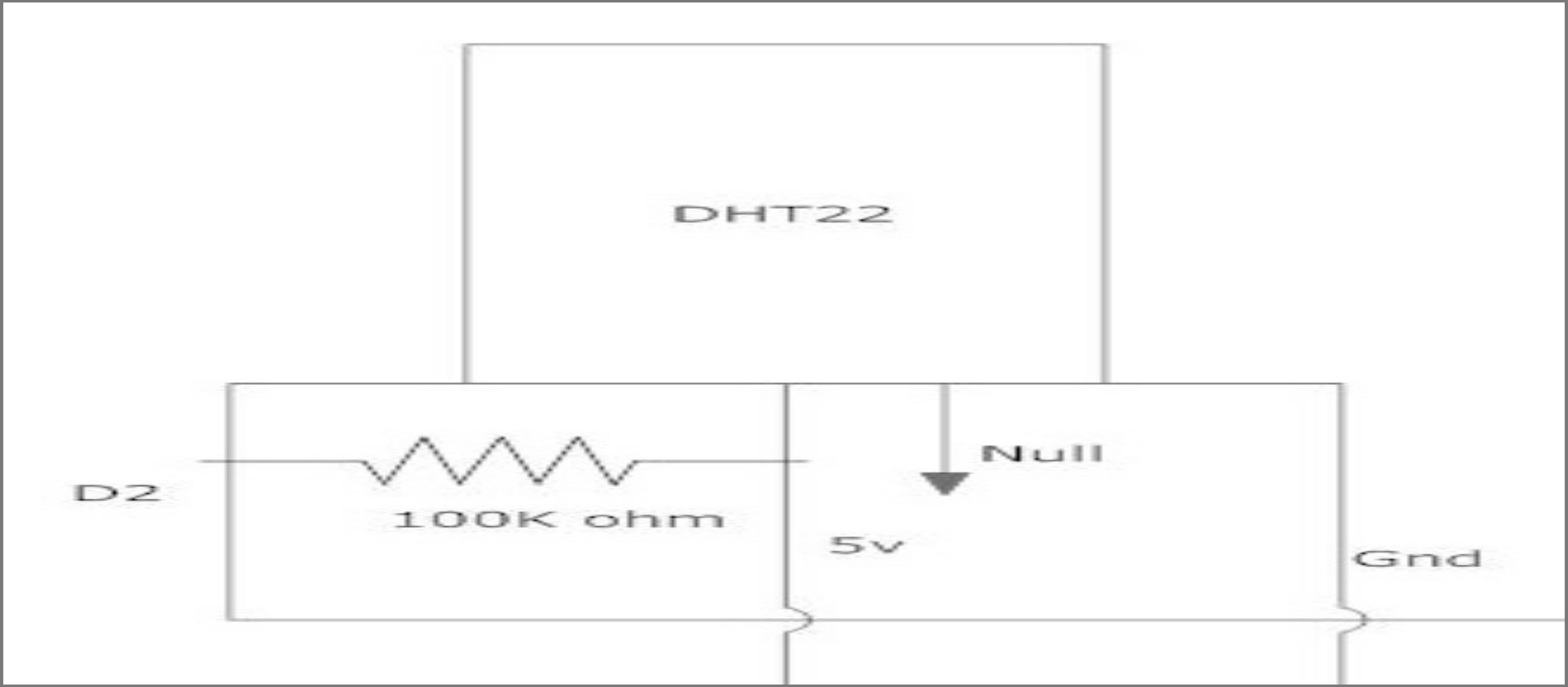
SD Card

- It’s 8GB SanDisk microSD card
- We are using an SD card adaptor so it can connect properly with data shield’s SD card slot.
- It is waterproof, shockproof, and X-ray proof
- It can operate in temperatures ranging from -13 to 185 degrees Fahrenheit,

DHT22 Temp/Humidity Sensor



DHT22 Temp/Humidity



- Pin1 – 5 volts
- Pin2 – Data pin from the Digital pin 2 and a pull up resistor is used from the voltage
- Pin4 – Ground pin

DHT22 Temp/Humidity

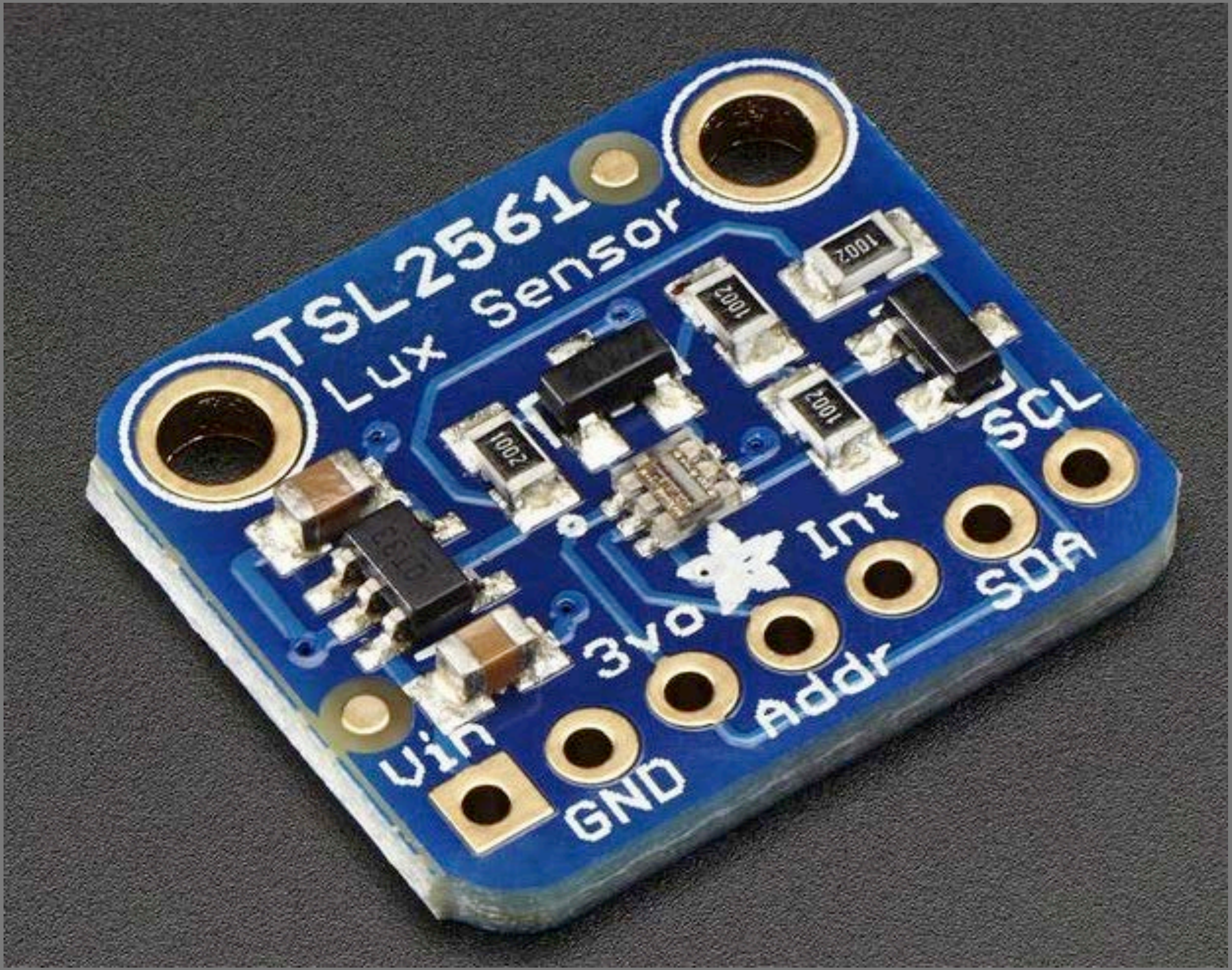
- 2.5mA max current draw
- Data is collected every two seconds
- Measures temperature (in fahrenheit and celsius), and humidity.
- Good for -40 to 125°C temperature readings
- Good for 0-100% humidity readings with 2-5% accuracy

DHT22 Temp/Humidity

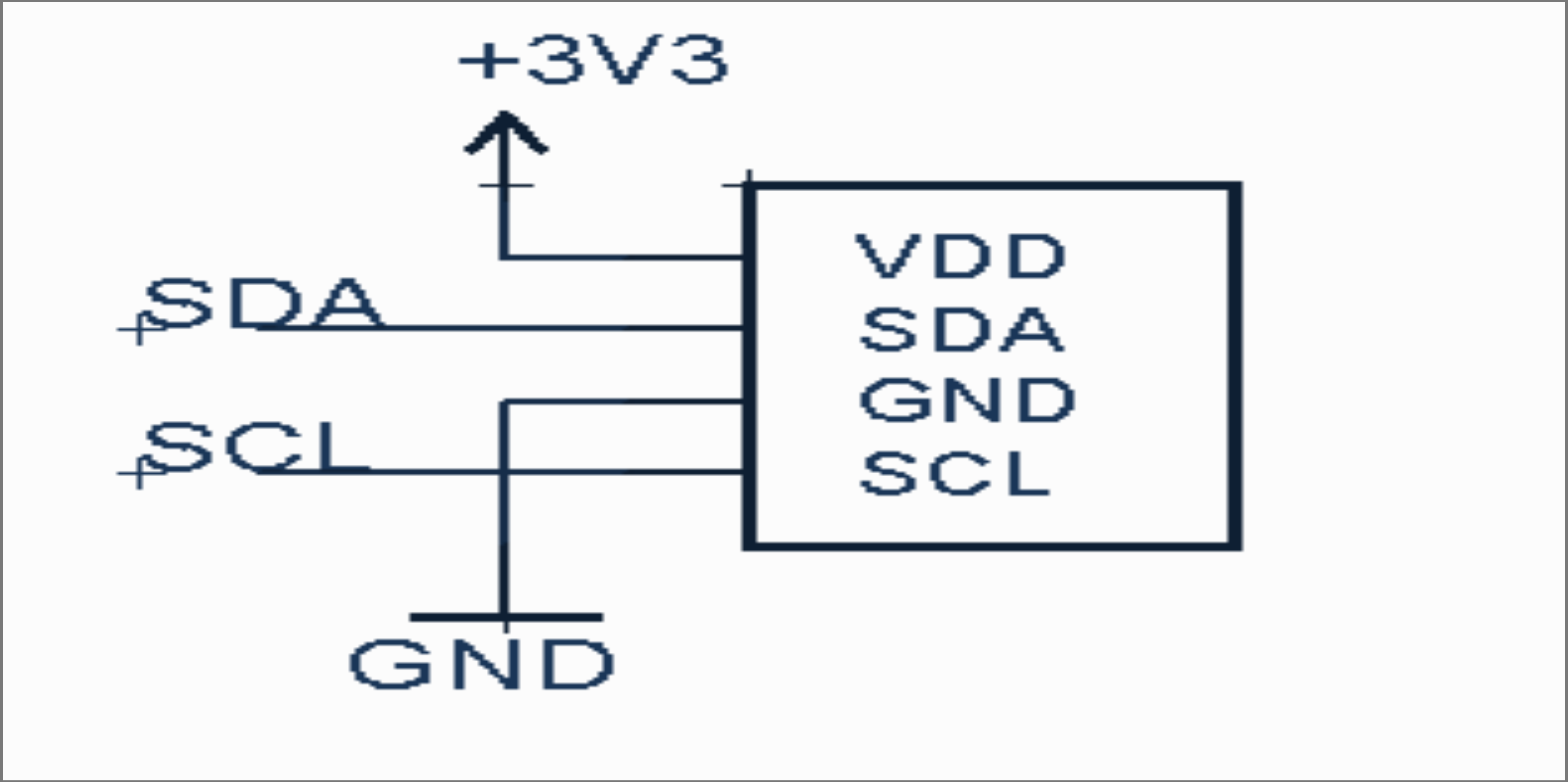
Temp/humidity sensor is used to measure internal temperature of the payload, since each component has operating and survival temperature range.

If the temperature is too cold then some components may not function, if temperature is too hot then some components may even be damaged.

Infrared Sensor



Infrared Sensor



- Vin – (Voltage in) takes in 3 volts
- SDA – (Data pin) to analog #4
- SCL – (Clock pin) to analog #5

Infrared Sensor

- Precisely measures illuminance in diverse lighting conditions
- Measures the amount of light present in lux
- New data is recorded every 250 milliseconds

Infrared Sensor

- Interface: I2C
- Temperature range: -30 to 80 °C
- Dynamic range (Lux): 0.1 to 40,000 Lux
- Wavelength range: 0.74 um - 1 mm

Infrared Sensor

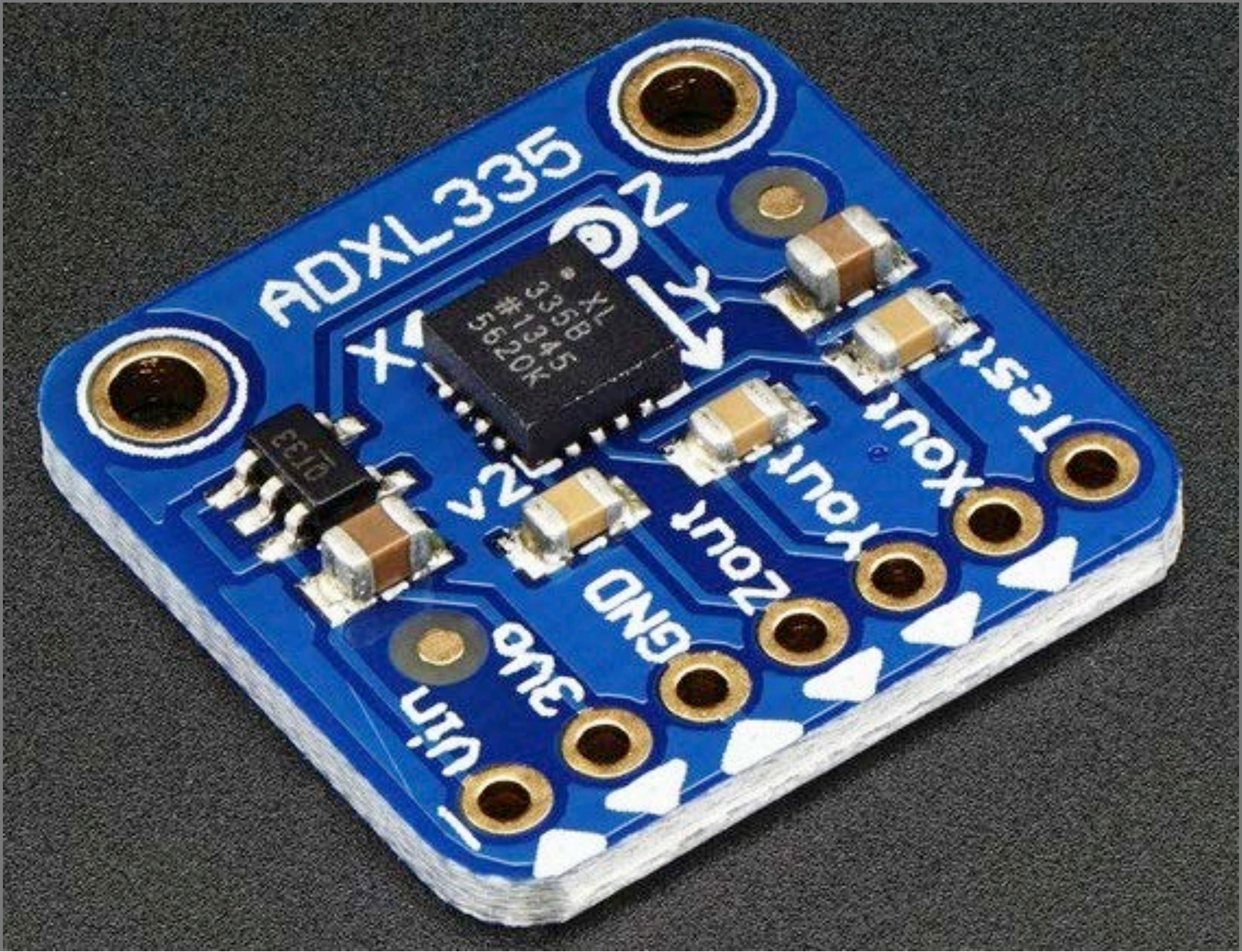
When we tested the sensor outside we discovered that all the readings came back wrong.

The sensor has the ability to change the sensitivity of the diodes automatically.

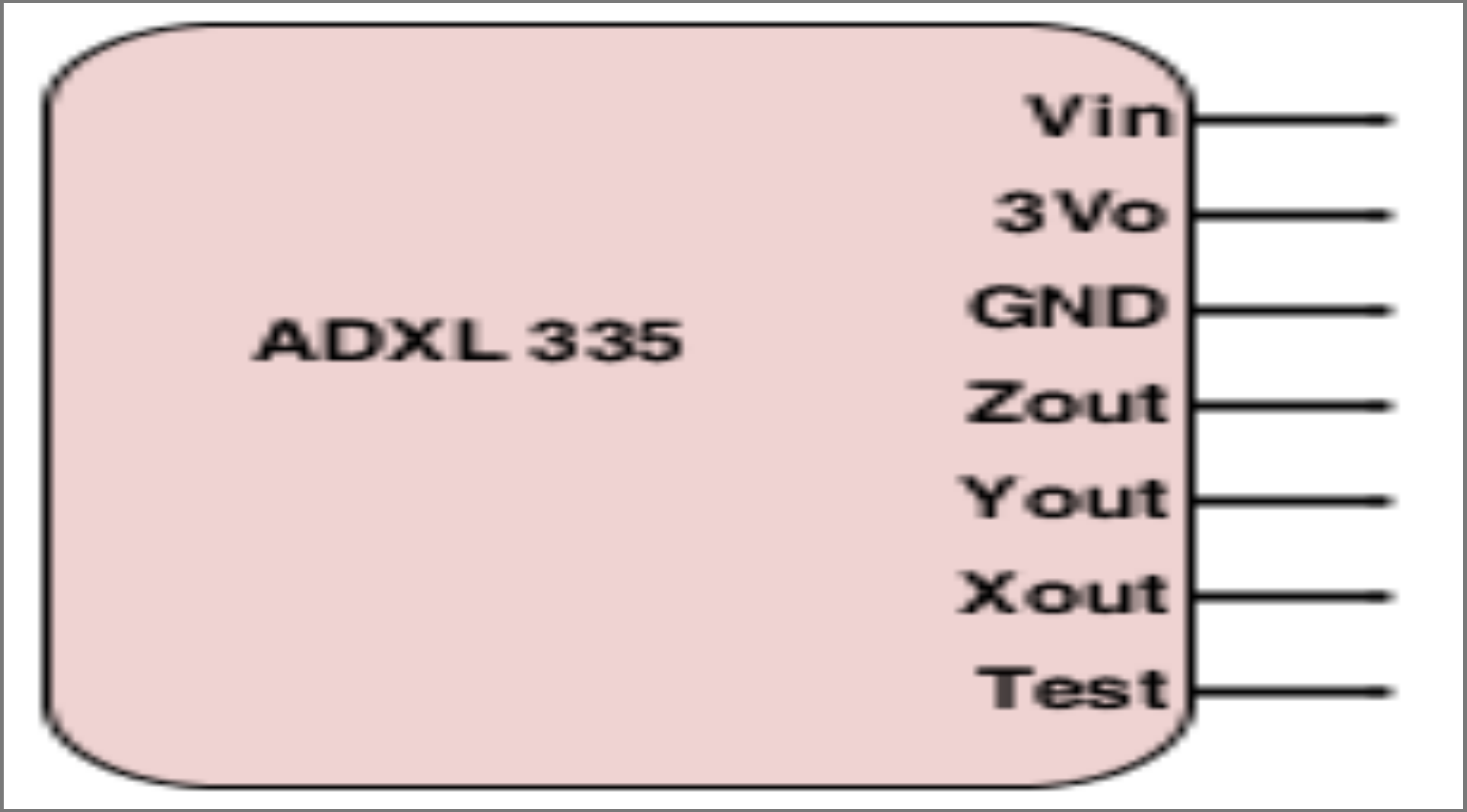
The sensor was overloaded by the brightness of direct sunlight.

Since the sensor didn’t change automatically, we modified a function in order to change it manually ourselves.

ADXL335


ADXL335
Z
Y
X
XL
335B
#1345
5620K
v2
Vin
3Vo
GND
Zout
Yout
Xout
Test

ADXL335



Vin – 5 volts GND – Ground

Zout – Analog #1 Yout – Analog #2 Xout – Analog #3

ADXL335

The ADXL335 is an accelerometer. They can measure the force of acceleration, whether caused by gravity or by movement.

For example in smartphones they allow you to adjust your screen by tilting your phone.

A 3-axis accelerometer that measures the acceleration of gravity on the x, y, and z-axes.

The range of the acceleration is from -3g to +3g.

ADXL335

- 4 mm × 4 mm × 1.45 mm LFCSP
- Low power - 350 µA (typical)
- Single-supply operation 1.8 V to 3.6 V
- Bandwidth adjustment with a single capacitor per axis

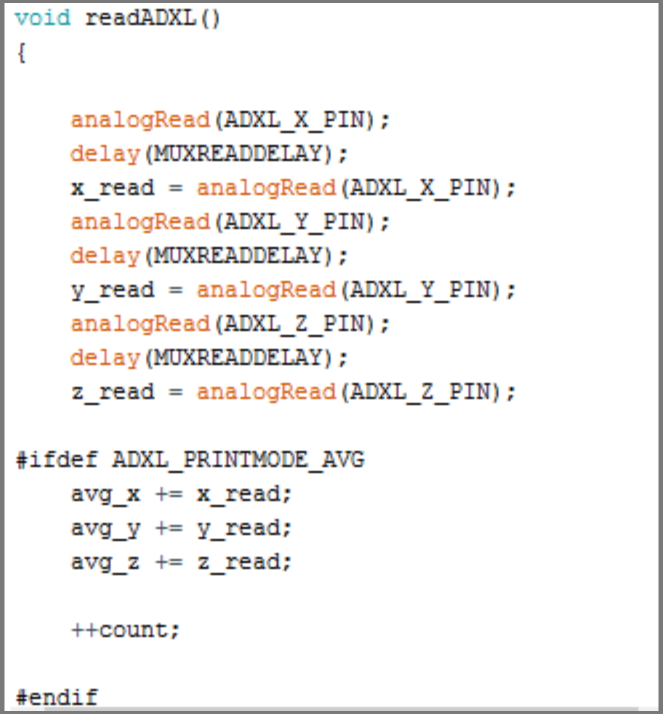
ADXL335 - Method and Testing

The accelerometer outputs from 0 to ~3.3 V and this is converted to a value from 0 to 1023 by our microcontroller’s analog to digital converter.

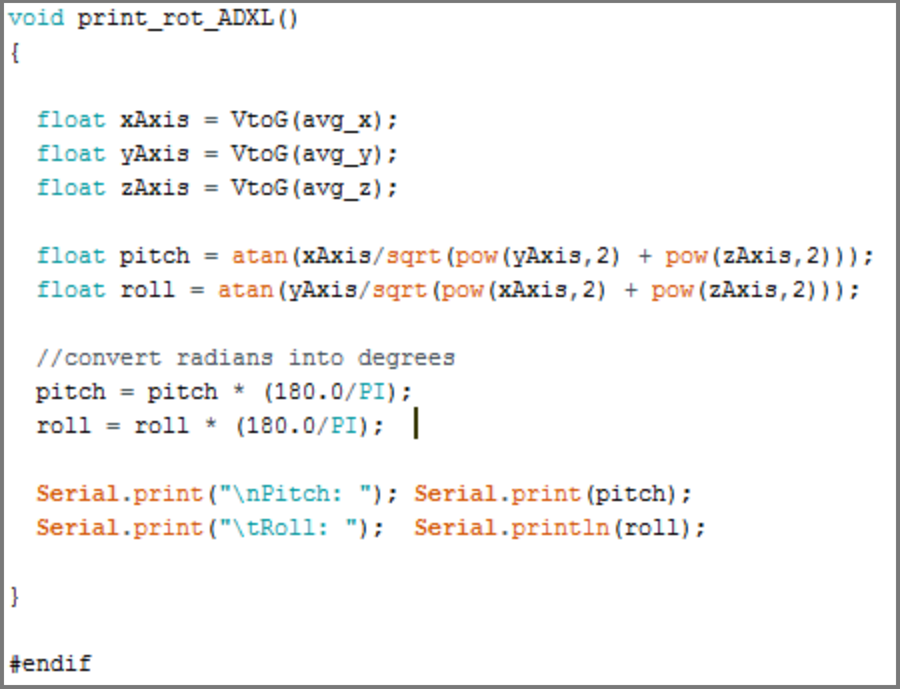
We wanted to convert this to G Forces so we took the raw ADC values and used a function to get our G forces.

To test this we created another function that used the G forces to calculate the roll and pitch of the sensor.

ADXL335 Code Snippet



```
void readADXL()
{

    analogRead(ADXL_X_PIN);
    delay(MUXREADDELAY);
    x_read = analogRead(ADXL_X_PIN);
    analogRead(ADXL_Y_PIN);
    delay(MUXREADDELAY);
    y_read = analogRead(ADXL_Y_PIN);
    analogRead(ADXL_Z_PIN);
    delay(MUXREADDELAY);
    z_read = analogRead(ADXL_Z_PIN);

#ifdef ADXL_PRINTMODE_AVG
    avg_x += x_read;
    avg_y += y_read;
    avg_z += z_read;

    ++count;

#endif
```

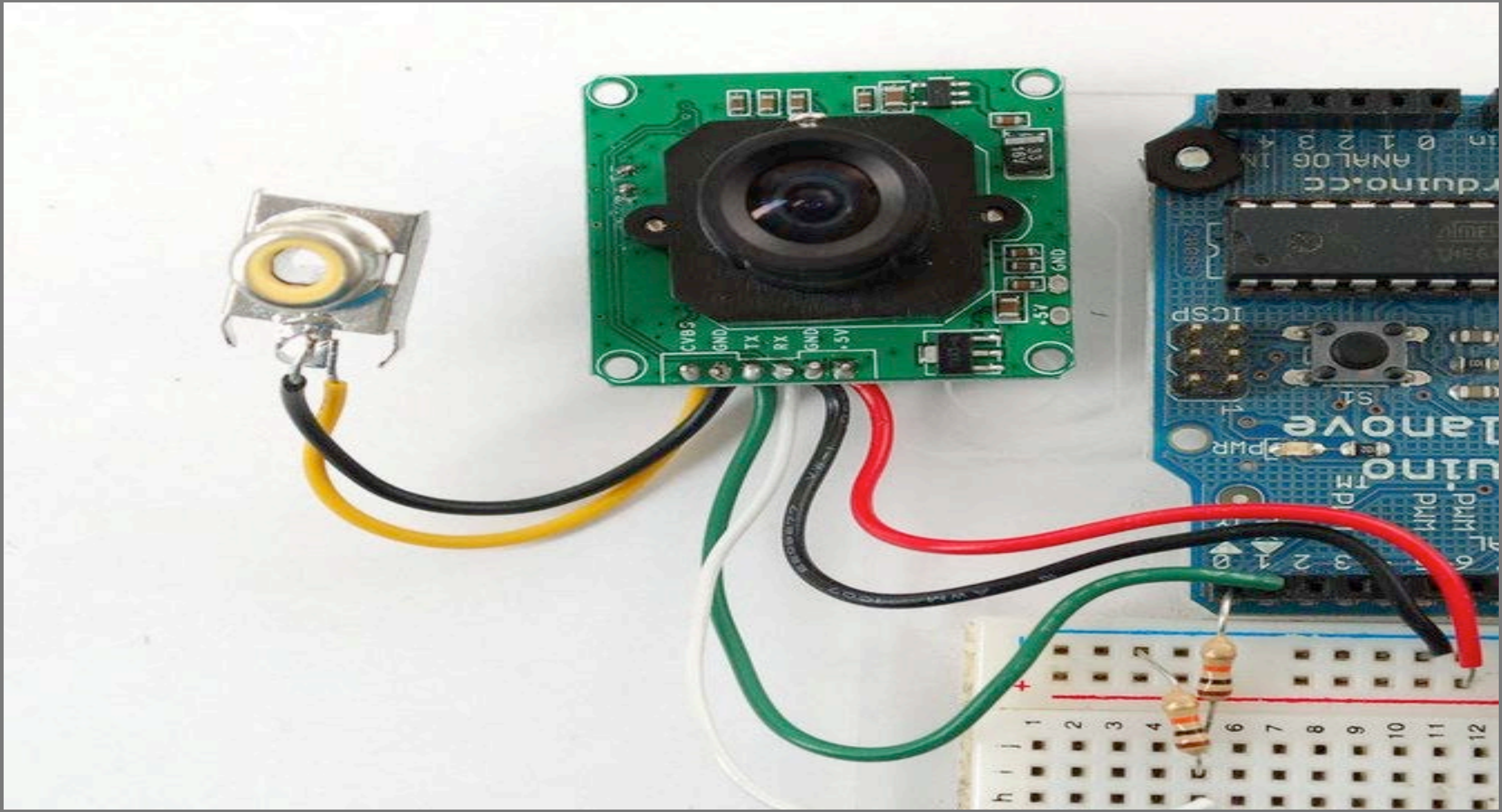


```
void print_rot_ADXL()
{

  float xAxis = VtoG(avg_x);
  float yAxis = VtoG(avg_y);
  float zAxis = VtoG(avg_z);

  float pitch = atan(xAxis/sqrt(pow(yAxis,2) + pow(zAxis,2)));
  float roll = atan(yAxis/sqrt(pow(xAxis,2) + pow(zAxis,2)));

  //convert radians into degrees
  pitch = pitch * (180.0/PI);
  roll = roll * (180.0/PI);

  Serial.print("\nPitch: "); Serial.print(pitch);
  Serial.print("\tRoll: ");  Serial.println(roll);

}

#endif
```

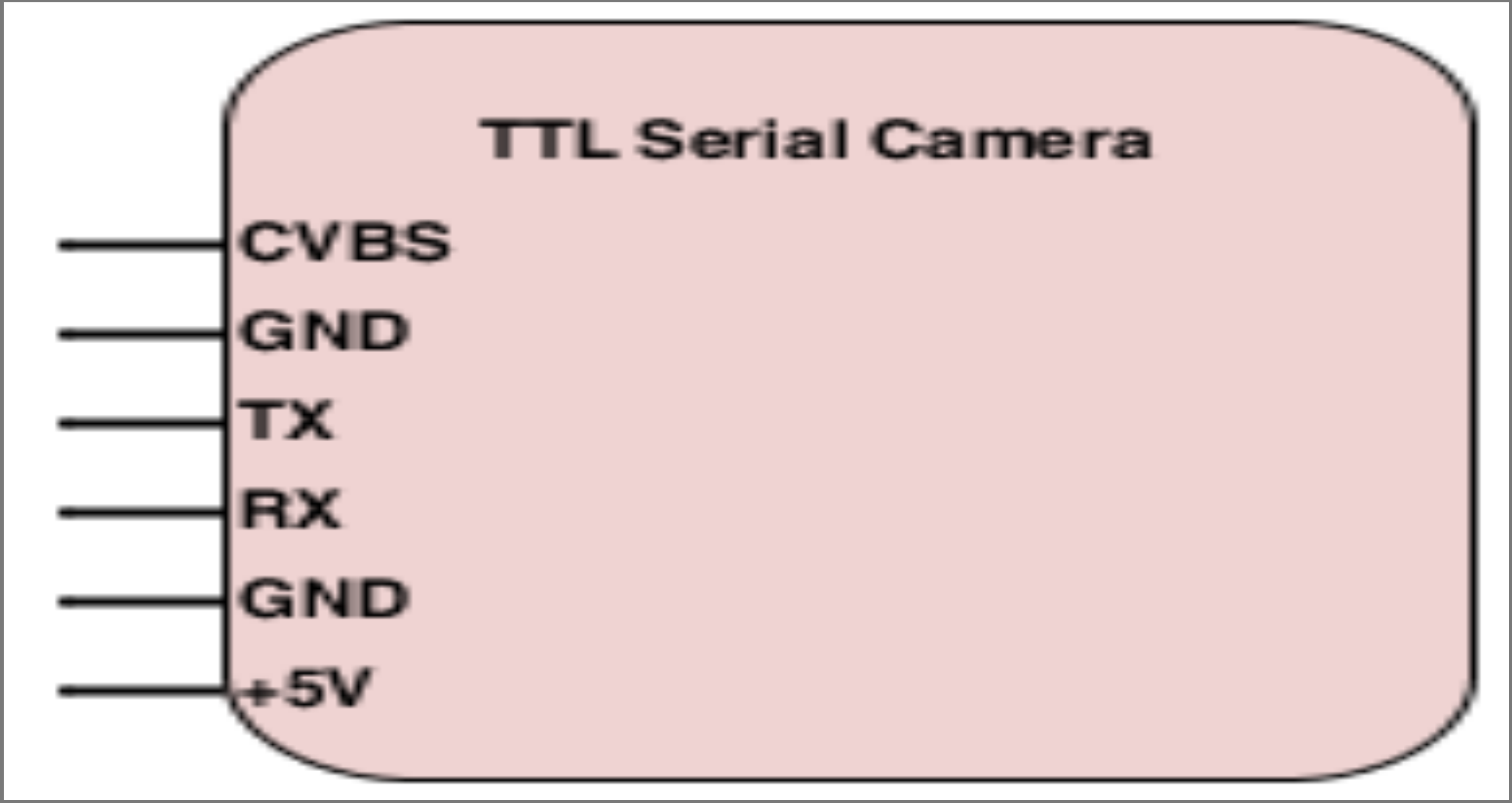
This part of the code is responsible for reading the x, y, and z values of the accelerometer

This part of the code was useful in testing our ADC to G-Force formula.

TTL Serial JPEG Camera



TTL Serial JPEG Camera



Uses 5 volts
Tx: connected to pin 1
Rx: connected to GND and pin 0
Second GND and CVBS connected to a RCA adapter

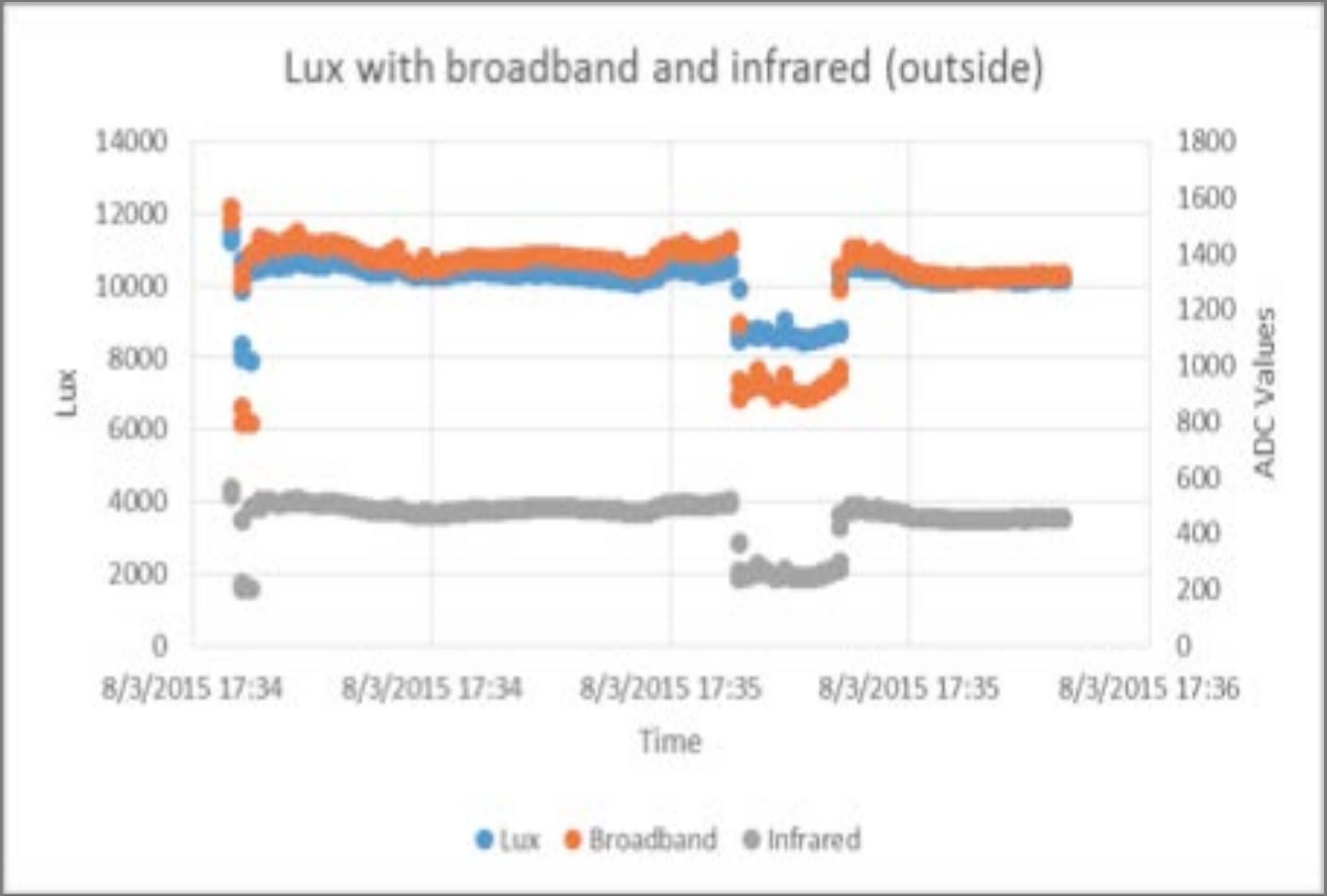
TTL Serial JPEG Camera

This camera records videos and take snapshots of these videos. Then it transmits them over the TTL serial link.

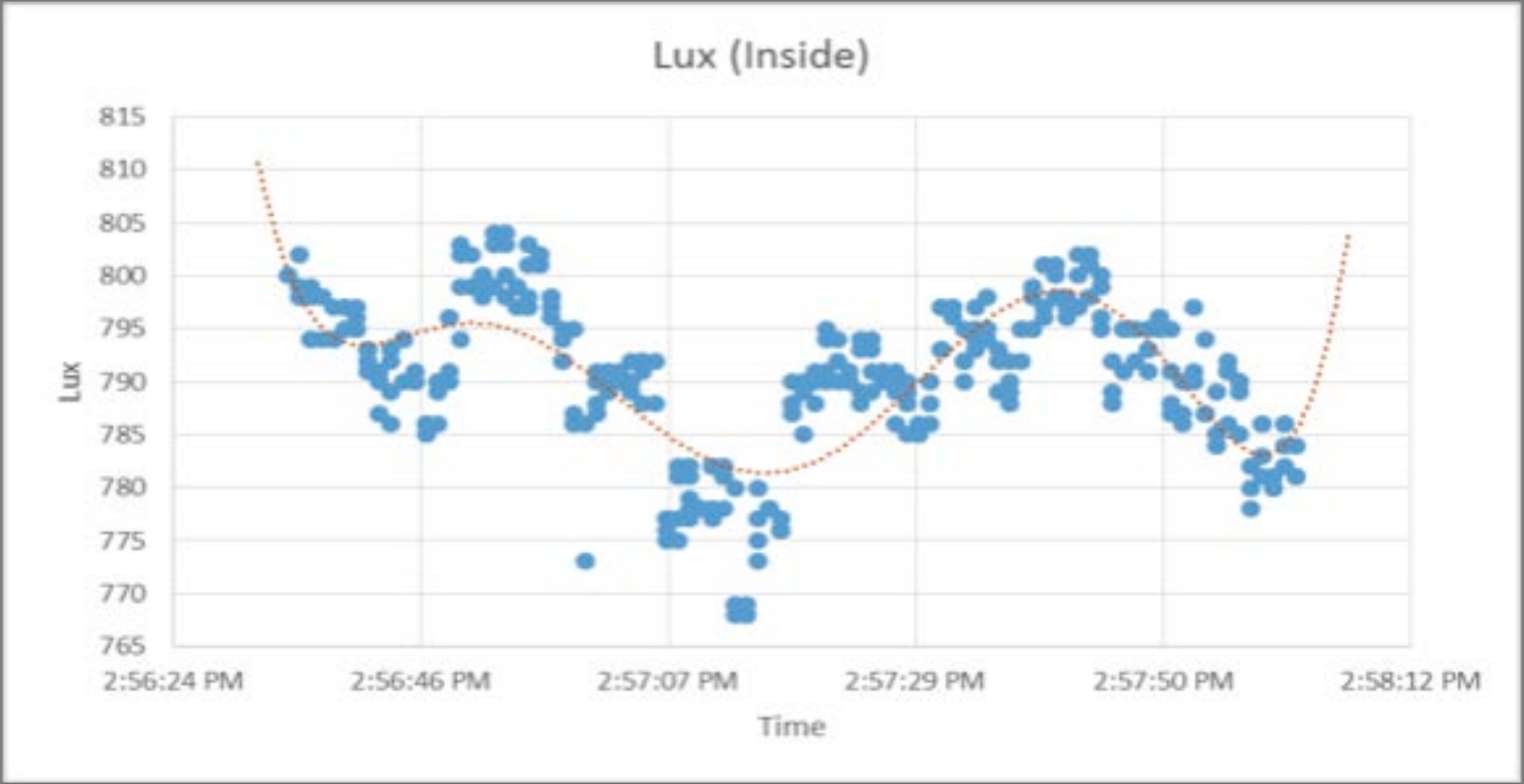
It has manually adjustable focus, auto-white-balance, auto-brightness and auto-contrast.

TTL Serial JPEG Camera

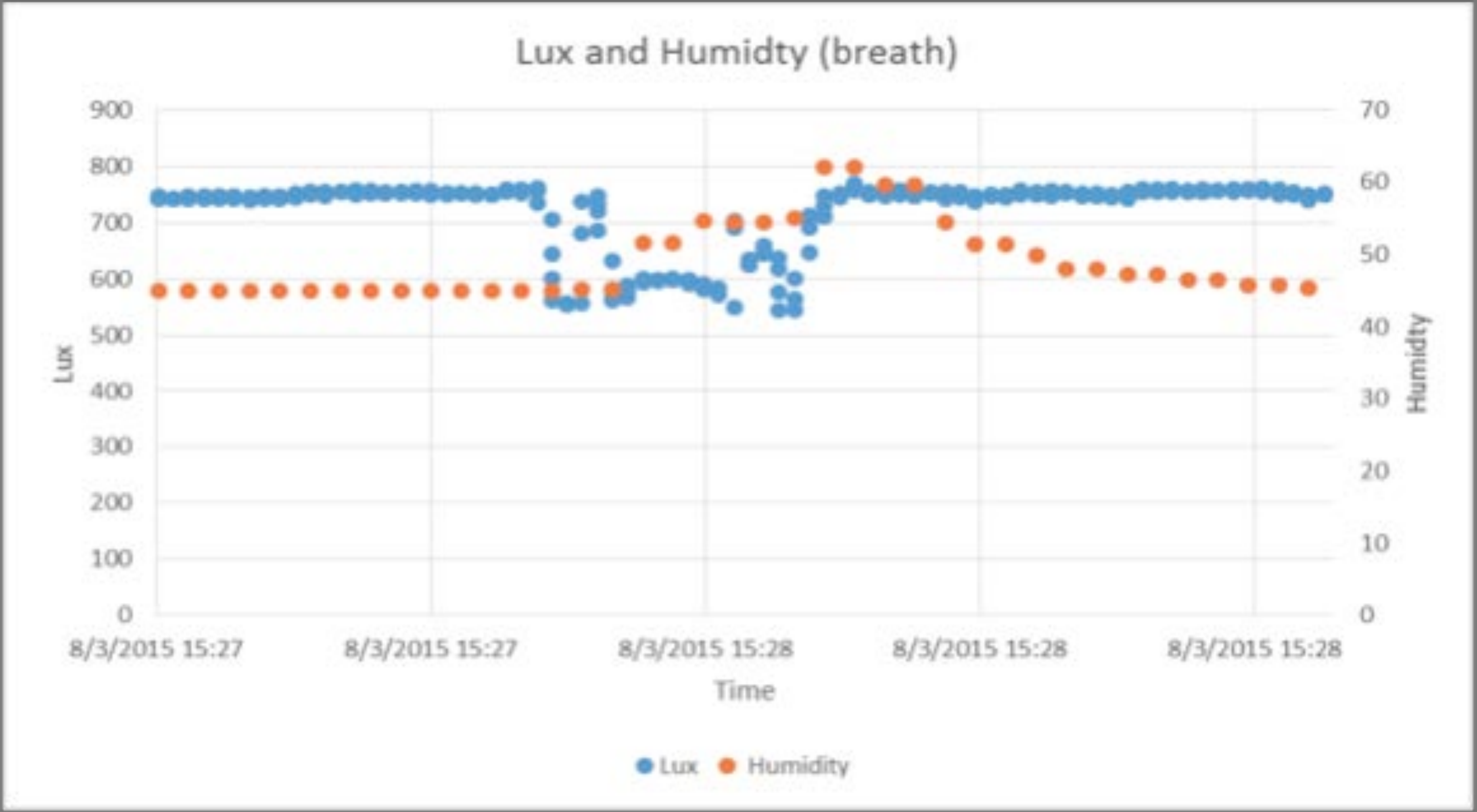
- Output format: Standard JPEG
- Frame speed: 30fps
- Resolution: 640*480 pixel
- Pixel size: 5.6um*5.6um
- Current draw: 75mA



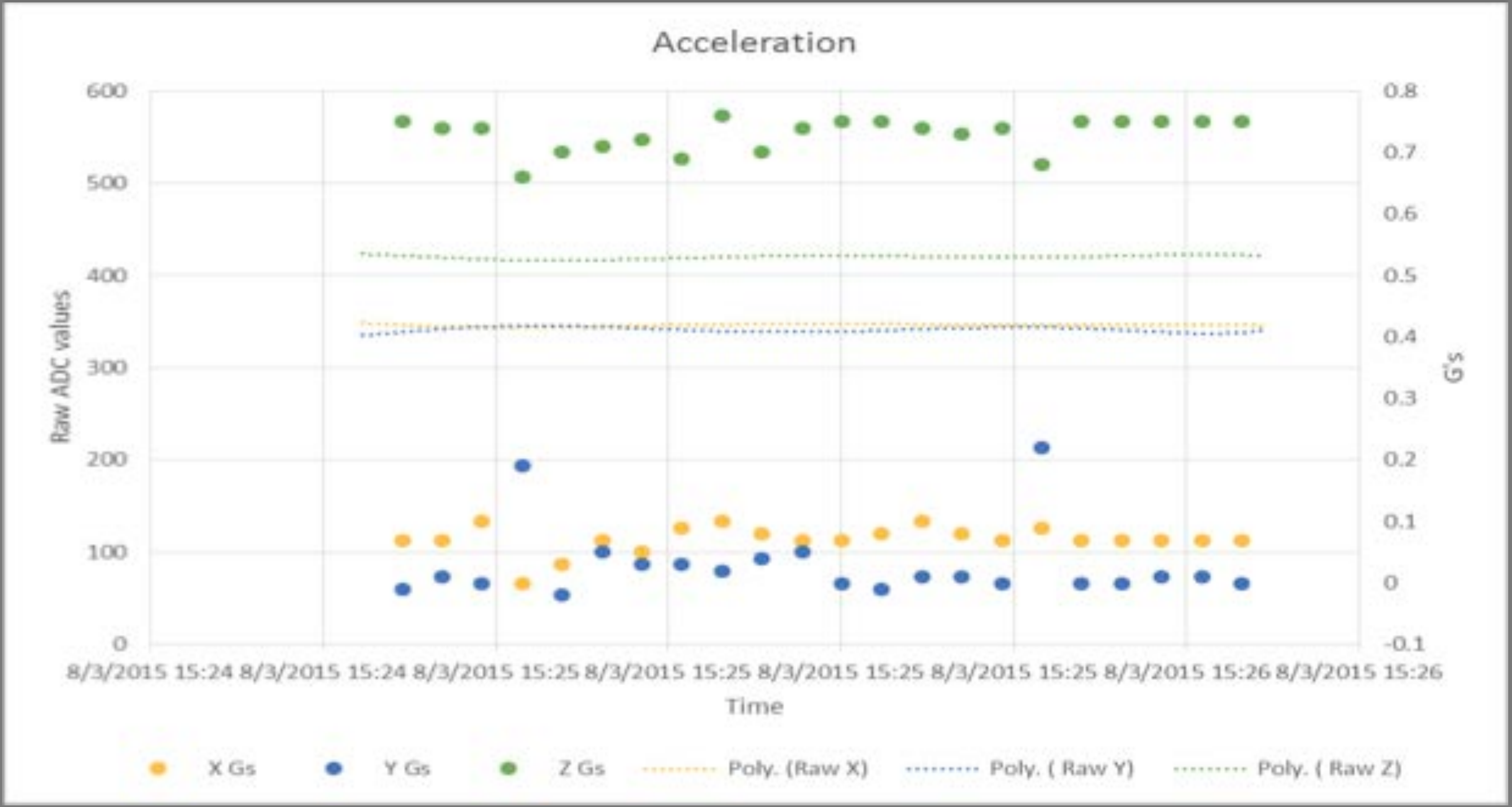
This figure represents light data taken outside of our college building.



This figure represents the lux calculated inside of a building. There is an unexpected sinusoidal trend that we believe may represent this particular light source inside the building, dimming and brightening at pseudo-regular rates.



This figure details the calculated lux and the humidity when someone exhales on the sensor. The dip in lux represents when the person blocked some of the light to the light sensor. An increase in humidity can be seen occurring immediately after the event representing the breath



The ADC on the Arduino converts voltages to a value between 0 and 1023. These are then converted to g's (which is a force caused by acceleration similar to gravity factor) using a formula and knowledge of the unit obtained from the datasheet.

What We Learned/Gained

- How to successfully write and integrate code
- How to setup and integrate hardware
- What we need to do to create future Cubesat
- We improved our collaboration techniques.
- We gained valuable knowledge that would be useful for future endeavors.

Conclusion

It is premature to draw conclusions about the feasibility of implementing an AC system for CubeSat, and additional investigation is necessary and ongoing.

Currently the IR sensor, Temperature/Humidity, and Accelerometer are communicating well with the Data logging shield and the Arduino. We will be doing a balloon launch using this prototype soon.

Reference

1. www.adafruit.com
2. www.arduino.cc
3. http://www.nasa.gov/sites/default/files/files/ELaNa-II-Factsheet-508(1).pdf
4. http://www1.cuny.edu/mu/forum/2013/11/14/medgar-evers-college-set-to-launch-satellite-on-nasa-rocket/

Acknowledgements

NASA New York City Research Initiative
New York NASA Space Grant Consortium
Goddard Institute of Space Studies
GSFC Office of Education

Questions?